**Crypto and Network Security**

**Lab 3**

**Name: GAURAV SETTY**

**Email:** [**settgm01@pfw.edu**](mailto:settgm01@pfw.edu)

**Task 1 – ARP Cache Poisoning**

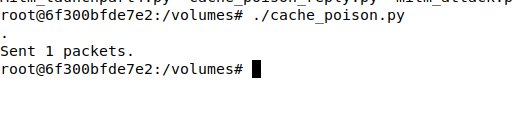
**Task 1.A: ARP Cache Poisoning using ARP Request**

**cache\_poison.py**

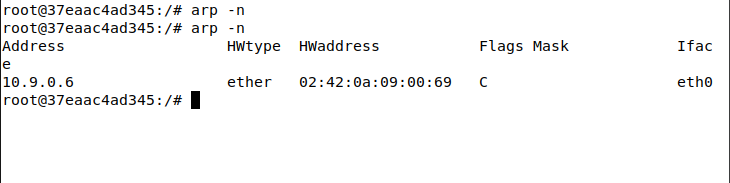


To trick host A into thinking the attacker's MAC address is associated with the IP address of B, an ARP request is made with an ether broadcast destination.

Running the code:



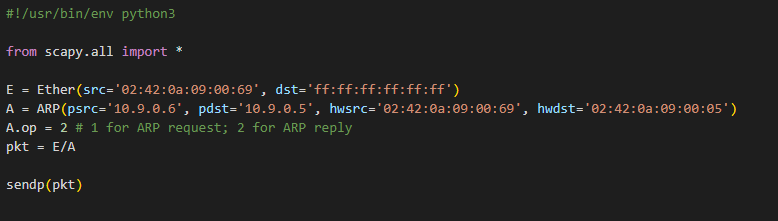
ARP cache of Host A before and after attack:



Host A’s cache is updated with the MAC address of the attacker mapped to Host B’s IP address.

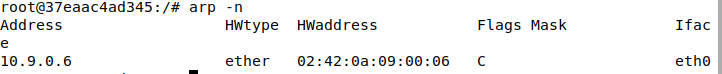
**Task 1.B: ARP Cache Poisoning using ARP Reply**

**cache\_poison\_reply.py**



To trick host A, an ARP reply is sent with B's IP address and M's MAC address.

**Case 1:** Host B’s IP address is already in Host A’s cache

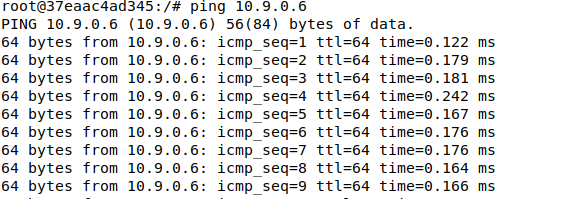


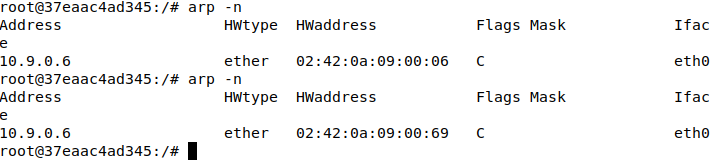
Host A’s cache points to Host B’s MAC address.

Running the code:



ARP cache of A after the attack:





The attacker's MAC address is updated in Host A's cache and assigned to Host B's IP.

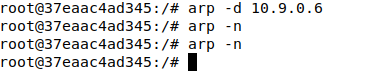
**Case 2:** Host B’s IP address not present in Host A’s cache

Clearing A’s cache:



Then run the cache\_poison\_reply.py code.

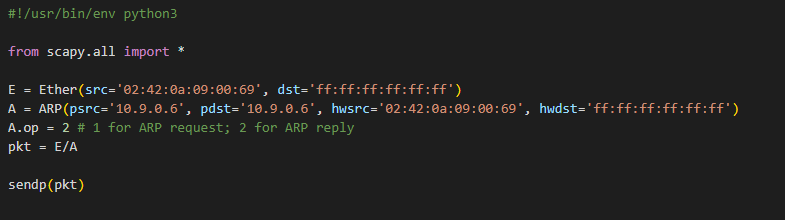
ARP cache of A after the attack:



The cache of A remains empty. As a result, the reply attack was unsuccessful since B's cache entry was missing.

**Task 1.C: ARP Cache Poisoning using ARP gratuitous message**

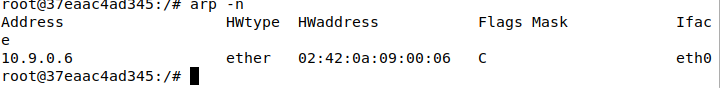
**gratuitous.py**



All network MAC addresses receive an ARP gratuitous message.

**Case 1:** Host A's cache already contains Host B's IP.

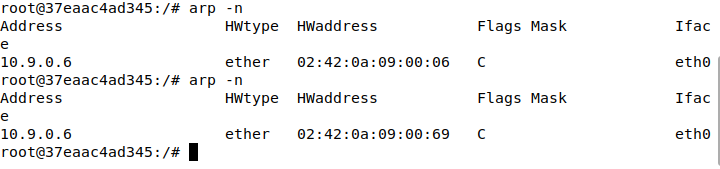
ARP cache of Host A before gratuitous message.



Running the code:



ARP cache of A after the attack:



The attacker's MAC address is now associated with B's IP address rather than B itself. Attack is effective.

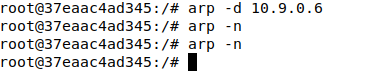
**Case 2:** Host B’s IP address not present in Host A’s cache

Clearing A’s cache:



Then run the gratuitous.py code.

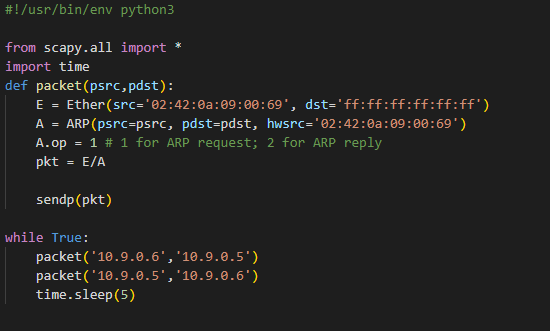
ARP cache of A after the attack:



The cache of Host A remains empty. This indicates that when host B is not listed in A's cache, the gratuitous request attack fails.

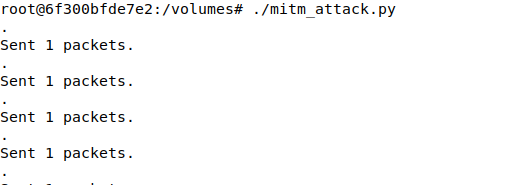
**Task 2: MITM Attack on Telnet using ARP Cache Poisoning**

**mitm\_attack.py**

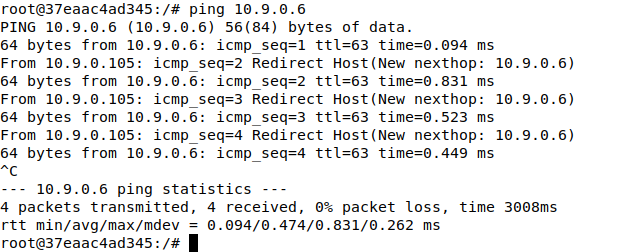


The code above sends out two phony ARP requests every five seconds: one to A with B's IP address mapped to M's MAC address and the other to B with A's IP address mapped to M's MAC address. As a result, M will receive all communications between A and B.

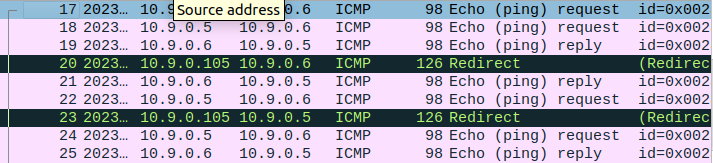
Running the code:



Now Pinging Host B from Host A:

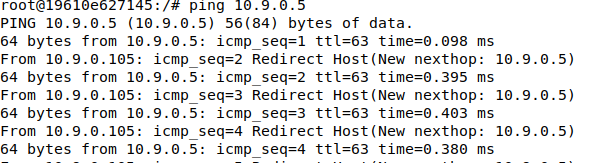


WireShark Observation:

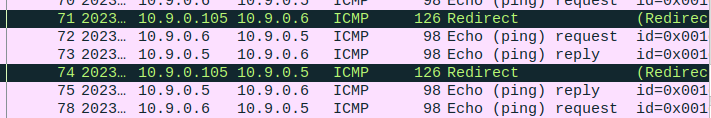


We can see the ping is being redirected from attacker’s machine (10.9.0.105).

Now Pinging Host A from Host B:

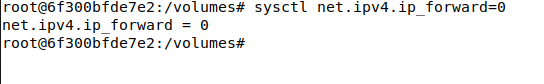


WireShark Observation:

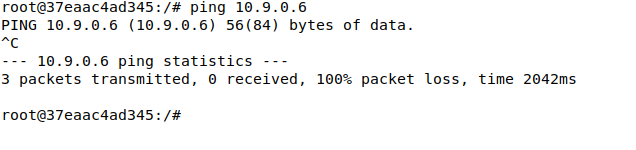


We can see that the attacker's system is forwarding the ICMP requests.

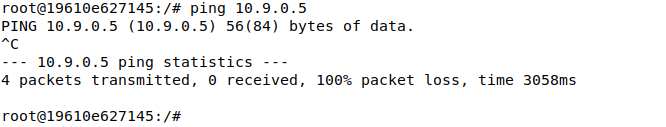
This is due to IP forwarding being enabled by default. Let's now disable IP forwarding:



Now Pinging Host B from Host A:



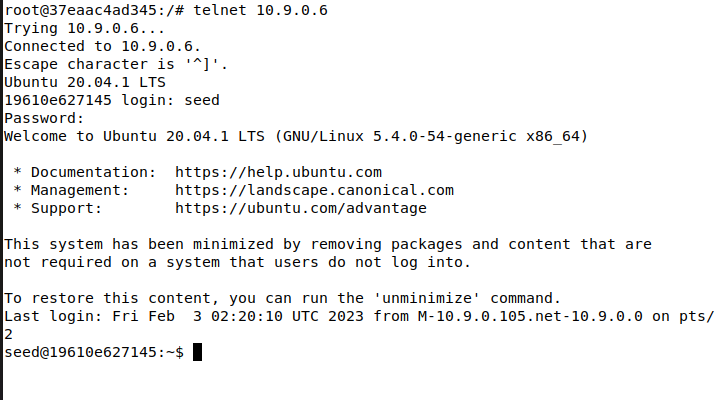
Now Pinging Host B from Host A:



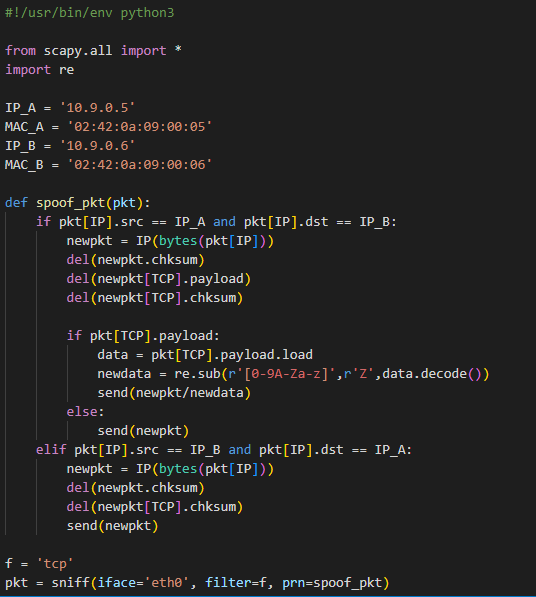
Due to the attacker's host M setting IP forward to 0, all packets were lost. From this, we should deduce that a potential MITM attack is occurring.

Attacking with the MITM:

First, we establish a telnet connection from A to B while the attacker machine's IP forwarding is turned on.

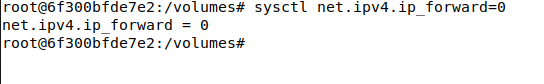


**Mitm\_launchpart4.py**

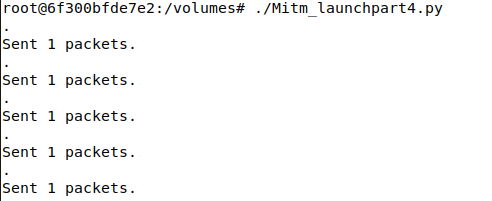


The skeleton code from Seed Labs is modified in the above code to replace all telnet characters with the letter Z. A substitution from the Python 3 regex package re is used to do this.

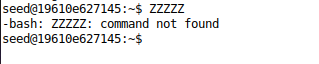
Now we set IP forward to 0 again:



Running the code:



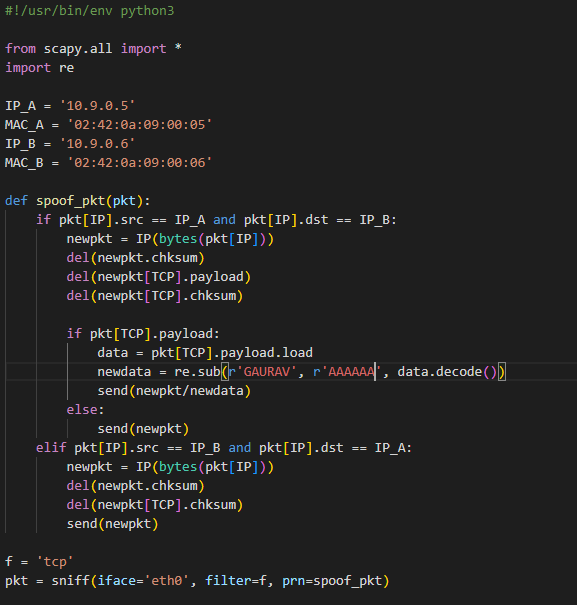
We can look at the telnet window and try typing something.



The letter Z is used in place of every character. The attack is going well. Each character is sent from A to B due to the way telnet is implemented, but the man in the middle (host M) intercepts these packets and substitutes the letter Z for the content. As a result, no matter what we enter, the letter Z always appears.

**Task 3: MITM Attack on Netcat using ARP Cache Poisoning**

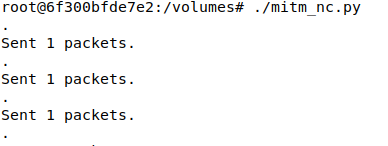
**mitm\_nc.py**



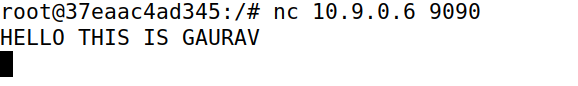
The letter A is substituted for my name in the code above.

The code mentioned above will now be executed to change my name in the netcat messages.

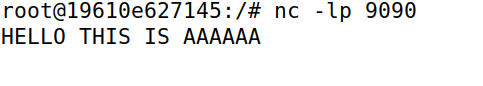
Running the code:



Sending messages from host A via netcat:



Receiving messages on host B from host A via netcat:



An effective Man In The Middle attack can be implemented by using an ARP cache attack in conjunction with sniffer and spoofing attacks. To do this, we create fictitious ARP requests in this lab and frequently transmit them to hosts A and B from the attacker host M. Next, we disable IP forwarding and begin spoofing any communications coming from either host, carrying out an MITM attack.